



Modelagem Lógica



A modelagem lógica é um importante passo no projeto de banco de dados. Nesta aula serão abordadas as principais características da modelagem lógica e como derivar um modelo conceitual para este.

CONCEITO DE MODELAGEM LÓGICA

Chegamos finalmente na modelagem lógica. Considerando as fases do projeto de banco de dados, a modelagem lógica sucede a modelagem conceitual e antecede a última fase, a física. Nessa fase o artefato gerado é um modelo lógico e pode-se dizer que ele é derivado do modelo conceitual.

Antes de começar essa modelagem, é muito importante entender algumas coisas que serão utilizadas. Primeiro é preciso relembrar o conceito de entidade e atributo. As entidades representam os objetos do mundo real, enquanto seus atributos representam as características destes objetos. Por exemplo, num sistema de e-commerce, você pode ter a entidade chamada "Produto", enquanto seus atributos possíveis podem ser "nome", "fabricante", "modelo", "descrição", "preço", entre outros.

Não podemos esquecer também que existem essencialmente dois tipos de atributos: comum (ou tipo "dado") e identificador (ou tipo "chave").

O atributo do tipo comum é aquele que não possui nenhuma restrição importante na modelagem conceitual e representa um dado normalmente, daquela entidade. Já o atributo do tipo chave é aquele que identifica o conjunto de dados. Em uma tabela um atributo tipo chave não pode ser nulo

nem mesmo se repetir nunca. Por exemplo, no mesmo cenário de um e-commerce, como cada produto possui características que podem s[illegible]petir, podemos criar um novo atributo chamado “id_produto” para considerá-lo como chave. Neste caso, este “id” é único de um produto (regra de nunca se repetir) e nunca podem existir produtos sem este dado (regra de nunca ser nulo).

O exemplo da tabela de produtos pode ser visto na figura 1, onde a modelagem conceitual deste simples objeto é descrita.

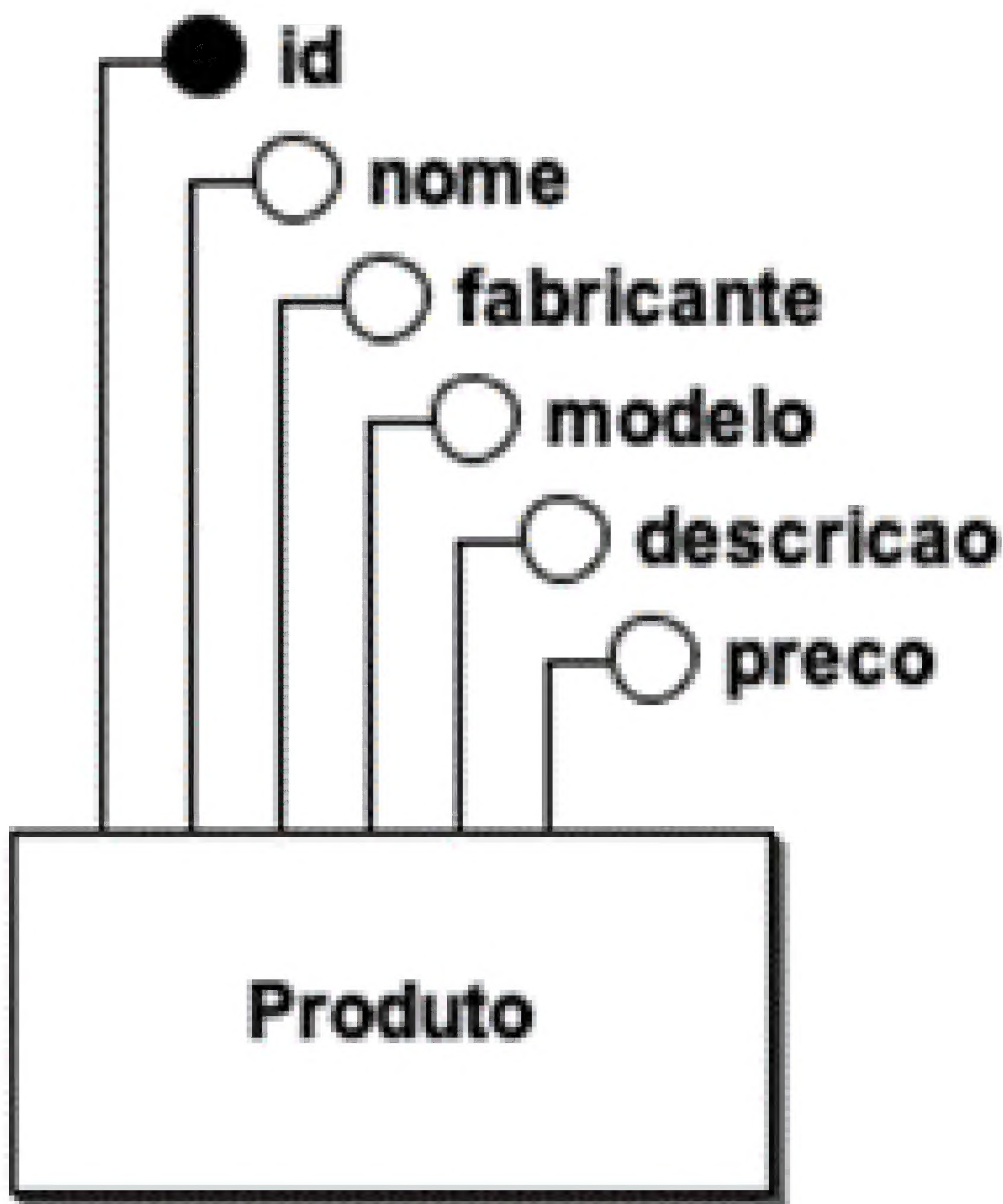


Figura 1 - Modelo conceitual da entidade "Produto" com seus respectivos atributos



Para ilustrar como ficaria a tabela deste exemplo, alguns dados arbitrários e fictícios podem ser vistos na tabela 1. Note que a tabela se chama "Produto" e cada um dos atributos descritos no modelo conceitual se tornaram as colunas. Note, também, que nessa tabela de exemplo os nomes das colunas estão todos em letras minúsculas e sem quaisquer acentuações ou caracteres especiais. Embora seja apenas um exemplo, já está sendo exibido no padrão lógico. Note, também, que a coluna "id" possui uma cor diferente para representá-la como chave.

Produto					
Id	nome	fabricante	modelo	descrição	preço
1	Discman Legal	DiscLegal	DM2020	Discman moderno e portátil	R$ 100,00
2	Walkman	WalkMusic	WM2020	Walkman retro com rebobinagem automática	R$ 80,00
3	MP4 player	MusicPlayer	MP42020	MP4 player com tela de 5 polegadas	R$ 200,00
4	MP4 Player	MusicPlayer	MP32020	MP3 player portátil estilo pendrive	R$ 400,00
5	Fone sem fio	MusicPlayer	FONBLU21	Fone de ouvido bluetooth com graves ótimos	R$ 1000,00

Tabela 1 - Exemplo de tabela de produtos

Lembre-se que a tabela anterior contém apenas exemplos arbitrários e está sendo usada apenas para referência. Note uma coisa importante nessa tabela: Cada coluna possui um tipo de dados específico. Isso é algo que devemos prestar bastante atenção na hora da modelagem lógica, pois os tipos de dados dependem da tecnologia de banco de dados que será utilizada. É por isso que durante a modelagem lógica já é preciso saber qual tecnologia de banco de dados será utilizada. Para todos os exemplos a partir de agora, será considerado o Oracle Database como foco e, por isso, toda a modelagem será baseada nele, inclusive realizada com ferramentas da própria Oracle, a partir de agora.

TIPOS DE DADOS



No exemplo mostrado anteriormente as colunas possuem tipos específicos de dados. Se estivéssemos olhando do ponto de vista de uma linguagem de programação, saberíamos que estes atributos são, do tipo: “id” é um inteiro, “nome”, “fabricante”, “modelo” e “descricao" são do tipo texto (String para muitas linguagens) e “preco” do tipo número real, ou float (variáveis de ponto flutuante) para muitas linguagens. Entretanto, no mundo dos bancos de dados, existem tipos específicos para se representar cada tipo em cada tecnologia utilizada. De forma resumida, a tabela 2 mostra os principais tipos de dados em bancos de dados Oracle, mas muito comuns, também em outros bancos de dados como, MySQL, MariaDB, SQL Server etc.).

Tipo	Descrição
CHAR(n)	Representa uma cadeia de caracteres. Se o “n” (tamanho) não for informado, representará apenas um único caractere. O tamanho máximo é 2000. Quando um dado não atinge o tamanho máximo uma cadeia de espaços em branco é gerada.
VARCHAR2(n)	Representa um texto, onde n pode ser de até 4000 caracteres. Quando um dado é inserido sem ocupar o tamanho máximo, o banco de dados não acrescenta espaços em branco.
NUMBER(n)	Representa um número de até “n” dígitos. Se o “n” não for informado, possuirá uma precisão de 38 dígitos significativos.
NUMBER(n, d)	Representa um número decimal de até “n” dígitos e precisão “d”, onde “d” é o número máximo de casas decimais após a vírgula.
DATE	Representa um dado do tipo data, com horas, minutos e segundos, de 1 Jan de 4712 AC até 31 de Dez de 9999 DC
BLOB	“Binary Large Object”, pode armazenar até 4GB de texto binário. Normalmente utilizado para arquivos (imagens, vídeos etc.)
CLOB	“Character Large Object”, pode armazenar até 4GB de texto.
BOOLEAN	O Oracle não implementa nativamente o tipo boolean. É preciso utilizar algum mecanismo como um char(1) ou um varchar2(1) ou number(1).

Tabela 2 - Tipos de dados mais usados em bancos Oracle

É muito importante salientar que há muitos outros tipos de dados, que serão, neste material, explorados aos poucos e conforme a necessidade. Os tipos de dados mostrados na tabela 2 representam os mais comuns, que podem resolver a maior parte dos casos.

No caso do exemplo da tabela produto, podemos notar que o “id” pode ser do tipo “NUMBER(5)”, ou seja, um número de até 5 dígitos enquanto os utos “nome” pode ser um VARCHAR2(40), ou sejam um texto de até quarenta caracteres, o “fabricante” pode ser tipo VARCHAR2(15), o “modelo” pode ser um VARCHAR2(30), a “descricao" pode ser um VARCHAR2(255) e, finalmente, o atributo “preco” pode ser um NUMBER(5,2), ou seja um número real de até 5 dígitos com duas casas decimais de precisão.

CHAVE ESTRANGEIRA (FK)

Na modelagem lógica um novo tipo de coluna aparece. As tão esperadas chaves estrangeiras ou, em sua nomenclatura original, foreign keys, as famosas FKs. É muito comum tratarmos esse tipo de coluna com o nome original em inglês ou, simplesmente, pela sigla dela.

As chaves estrangeiras são importantes pois são elas que representarão os relacionamentos entre as entidades (que agora começamos a chamar de tabelas). Sempre que estamos derivando o modelo conceitual para o lógico, teremos uma série de chaves estrangeiras em nosso modelo.

Conceito importante sobre chave estrangeira: Todas as colunas marcadas como “FK”, sempre apontam para uma coluna de chave primária (primary key – PK), ou seja, os dados que são inseridos dentro das colunas do tipo “FK” são dados que apontam para outros dados, sempre identificadores (chave) de alguma outra entidade (relacionada) com a tabela que está recebendo essa FK. Eventualmente apontam para a mesma tabela, em auto relacionamentos.

Para melhor entendimento, vamos ao exemplo original do “produto”, mas agora será acrescido uma nova tabela chamada “categoria”, conforme descrito pelo modelo conceitual da figura 2.



Figura 2 - Modelo conceitual de exemplo, com uma nova entidade

Neste exemplo, fora adicionado uma entidade chamada categoria, onde o produto deve possuir no mínimo uma categoria e no máximo uma categoria, também. Por outro lado, a categoria pode não conter nenhum produto, mas também pode conter vários produtos. Para o relacionamento em questão, uma FK deve ser gerada na tabela de produto, ou seja, uma nova coluna deve ser incluída na tabela de produto indicando a qual categoria ele pertence, mas através de seu “id_categoria”. Para ilustrar essa ideia, veja a tabela categoria implementada com dados arbitrários.

Tabela 3 - Exemplo de dados na tabela "categoria", do exemplo



Categoria	
id_categoria	departamento
100	Tocadores de CD
101	Tocadores de Fita K7
102	Tocadores Digitais
103	Fones
104	Baterias

Isso é uma FK!

Tabela 4 - Tabela de produtos com relacionamento para categoria

Produto						
Id	nome	fabricante	modelo	descrição	preço	Id_categoria
1	Discman Legal	DiscLegal	DM2020	Discman moderno e portátil	R$ 100,00	100
2	Walkman	WalkMusic	WM2020	Walkman retro com rebobinagem automática	R$ 80,00	101
3	MP4 player	MusicPlayer	MP42020	MP4 player com tela de 5 polegadas	R$ 200,00	102
4	MP4 Player	MusicPlayer	MP32020	MP3 player portátil estilo pendrive	R$ 400,00	102
5	Fone sem fio	MusicPlayer	FONBLU21	Fone de ouvido bluetooth com graves ótimos	R$ 1000,00	103

Note que cada categoria, neste caso, possui um "id" que é único da categoria e, para satisfazer esse tipo de relacionamento, o "id_categoria" deve ser adicionado na tabela de produtos, conforme exemplificado com dados arbitrários da tabela 4. Isso quer dizer que a "id_categoria" na tabela de produtos é um FK para tabela "Categoria". Neste modelo foi adicionada a FK na tabela "protudo" pois cada produto possui uma categoria, ou seja, o entendimento do contexto também é necessário para a inclusão dessa FK no local correto e indicada pela modelagem lógica, vista a seguir.

MODELAGEM LÓGICA

Uma vez compreendidos os principais conceitos da modelagem lógica, os tipos de dados e os tipos de colunas (chave, dado e chave estrangeira), podemos finalmente começar a criar a modelagem lógica, que possui uma

notação bastante simples, mas regras importantes de derivação do modelo conceitual, apresentadas posteriormente.

A modelagem lógica é composta por um diagrama onde cada entidade é representada por um retângulo dividido em até quatro partes, horizontalmente. Na primeira parte (superior) coloca-se o nome da tabela (ou entidade). Na segunda parte coloca-se os atributos indicando-se os tipos ao lado direito, na terceira parte os atributos chaves e na quarta parte os atributos que são chaves estrangeiras.

A figura 3 mostra o diagrama do exemplo desta aula criado utilizando o software "Oracle SQL Developer Data Modeler" que é gratuito e bastante completo. É muito comum se utilizar softwares diferentes para a modelagem lógica e se obter resultados distintos ao mostrado abaixo, mas o importante é que todos irão representar as entidades de forma muito similar. Outros softwares que também realizam a modelagem lógica são o brModelo, Astah Professional (pago), StarUML (gratuito para testar), MySQLworkbench (para MySQL), site draw.io (online), site sqldbm.com (online e ainda não disponível para Oracle, embora em processo de atualização), entre muitos outros.

Note neste diagrama algumas peculiaridades. Na tabela de "Produto", em frente ao atributo "id" há um "P" que indica que este é um atributo chave (primary key) em frente ao "id_categoria" há um "F", que indica que este é um atributo do tipo "foreign key", que aponta pra a entidade "Cartegoria". No caso dos bancos e dados Oracle, as FKs recebem um registro, chamado de "constraint" no banco e elas recebem um nome. Neste caso esse relacionamento foi chamado de "Produto_Categoria_FK" e está aplicada à coluna "id_categoria". O pequeno asterisco (*) vermelho em frente à cada um dos atributos indica que eles não podem ser nulos, ou seja, são campos obrigatórios. Isso, em frente ao "id_categoria" da tabela "Produto" garante a cardinalidade mínima de 1, ou seja, todo produto obrigatoriamente deve conter uma categoria.

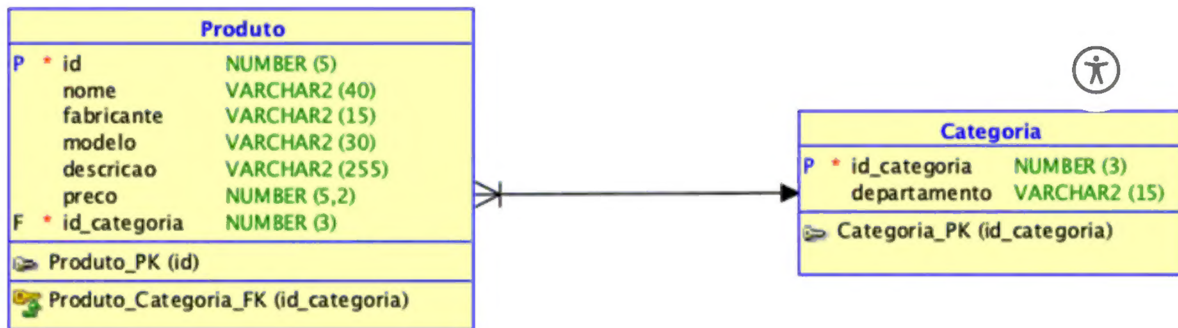


Figura 3 - Modelo lógico

Como mencionado anteriormente, existem várias formas de se representar a modelagem lógica. Veja, uma outra forma bastante usual, um exemplo um pouco mais completo do modelo conceitual, representado pela figura 4 e sua derivação na modelagem lógica, representada na figura 5. Tente analisar cuidadosamente como cada uma das entidades e seus respectivos relacionamentos foram derivados, especialmente a entidade associativa, onde gerou-se uma tabela especial apenas para este relacionamento.

Note, também, a linha que foi utilizada neste diagrama para representar a modelagem. Como, neste caso, a relação é “um para um” do lado de produto em relação à categoria e “zero para n” ao lado da categoria em relação ao produto, usa-se o símbolo de “pé de galinha” na tabela de produto e “seta” na tabela “categoria”. Não se preocupe, pois, essa simbologia ficará mais clara em breve. As regras de derivação entre a modelagem conceitual para a modelagem lógica são muito bem definidas e serão apresentadas em uma aula para este fim.

Como mencionado anteriormente, existem várias formas de se representar a modelagem lógica. Veja, uma outra forma bastante usual, um exemplo um pouco mais completo do modelo conceitual, representado pela figura 4 e sua derivação na modelagem lógica, representada na figura 5. Tente analisar cuidadosamente como cada uma das entidades e seus respectivos relacionamentos foram derivados, especialmente a entidade associativa,

onde gerou-se uma tabela especial (tabela de relacionamento) apenas para este relacionamento.

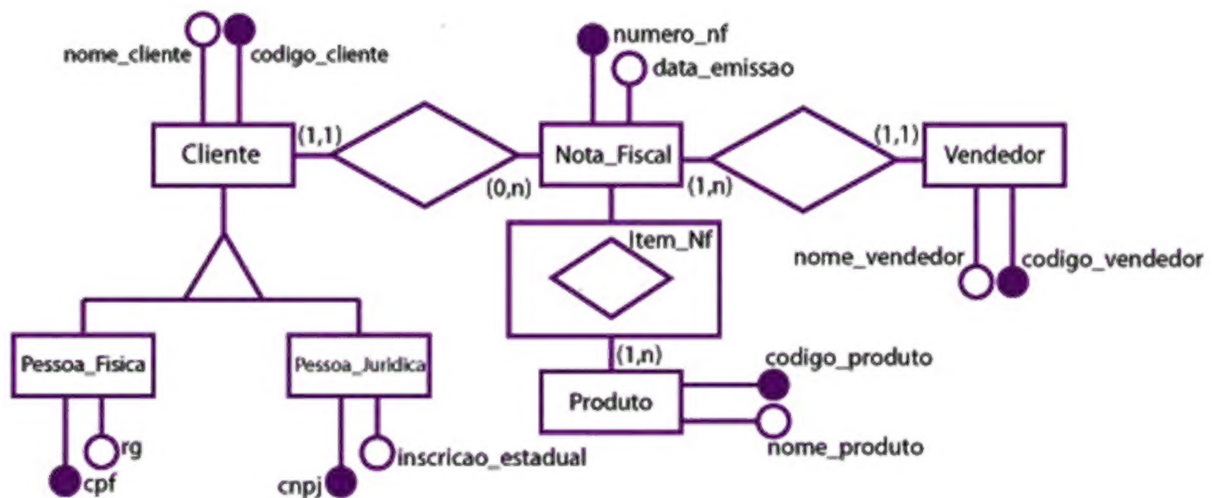


Figura 4 - Exemplo de modelagem conceitual

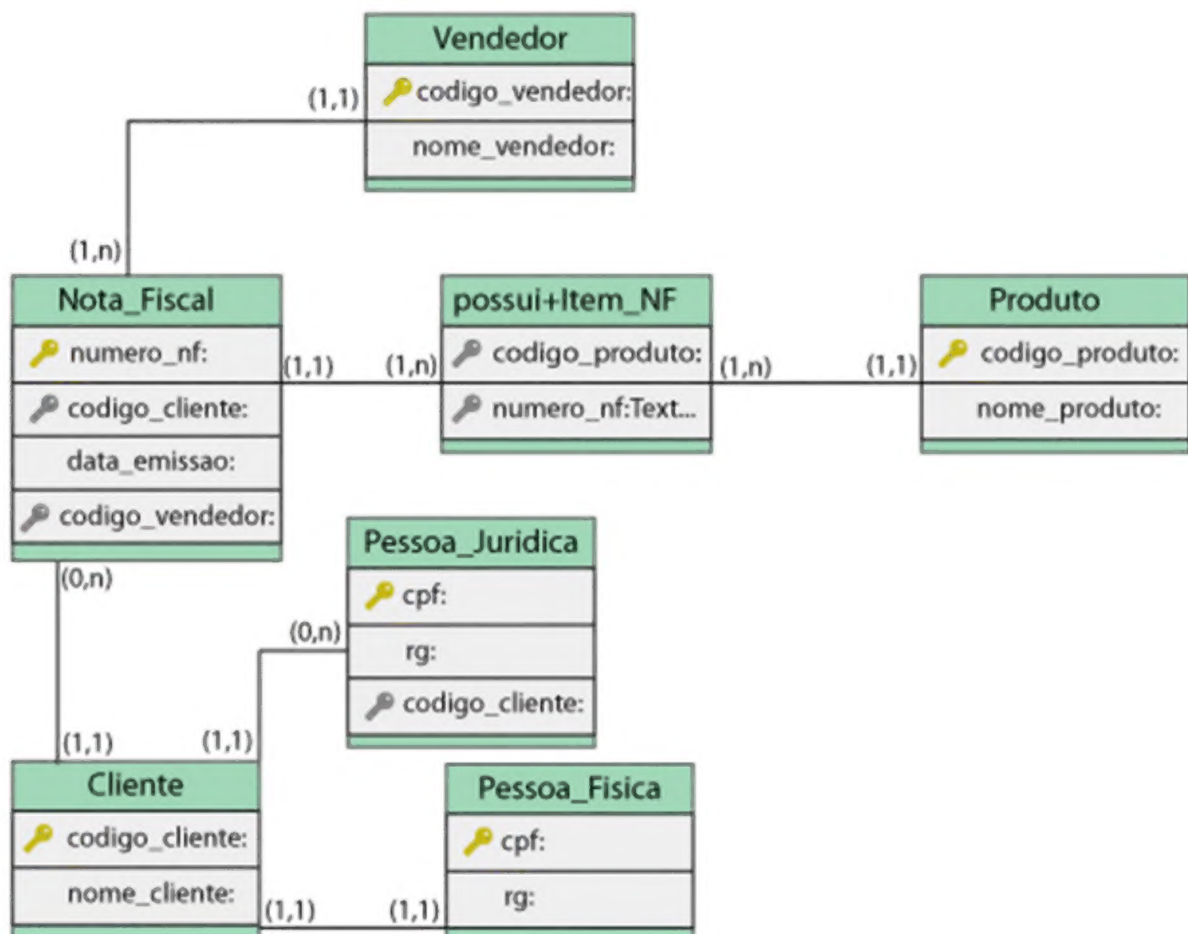


Figura 5 - Modelagem lógica derivada

ABORDAGENS DE ANÁLISE DOS MODELOS



A modelagem dos dados não é tarefa executada numa única iteração devido a sua natura e complexidade. Ela precisa ser feita gradativamente e, a cada etapa, são acrescidos novos itens aos já existentes, tornado o modelo gradativamente melhor. Na prática nenhuma das estratégias de análise e propostas na literatura é aceita de forma universal. Normalmente usamos construção dos modelos considerando três possíveis abordagens (ou metodologias) de modelagem de dados. (HEUSER, 2004).

Podemos realizar a modelagem dos dados através de várias fontes, como por exemplo a descrição textual de dados preexistentes (que já existem) ou de conhecimentos do mundo real modelado, relatado verbalmente pelos envolvidos no projeto. Neste caso, então usamos a engenharia reversa, também chamada de abordagem *bottom-up* (baixo para cima). Nesta abordagem iniciamos a modelagem utilizando tabelas e dados existentes e formatados em um banco de dados, apenas os adaptando até a modelagem conceitual.

Quando utilizamos a análise de requisitos do domínio (contexto) da aplicação, ou seja, o projeto modelado é desenvolvido a partir do que se conhece sobre o domínio modelado, podemos utilizar duas possíveis estratégias: top-down (cima para baixo) ou, ainda, o inside-up (dentro para fora) (HEUSER, 2004).

Na metodologia top-down, partimos da análise de requisitos para identificarmos as entidades do mundo real e criamos primeiro a modelagem conceitual, definimos os relacionamentos e suas respectivas cardinalidades. Em seguida, podemos realizar alguns testes de validação com dados fictícios que simulam a realidade e o usuário do banco pode, inclusive, participar deste processo.

Já na metodologia inside-up partimos de uma ideia centralizadora (central) onde definimos as principais entidades que fazem parte do mundo real e, em seguida, são incluídas ao centro do modelo. A partir dele, começamos a realizar os detalhamentos, ampliando gradativamente os relacionamentos, identificando as cardinalidades etc., como fazemos no processo top-down. Em uma primeira etapa, desenhamos o modelo conceitual com seus respectivos relacionamentos e cardinalidades, eventuais generalizações e especializações. Depois disso, definimos cuidadosamente os atributos comuns. Finalmente, na última etapa, realizamos os testes de validação, com dados arbitrários e onde o usuário final pode participar também.

Entender essas abordagens pode nos ajudar a entender qual a melhor forma de aprender o processo de derivação, pois cada pessoa é única e possui suas preferências de aprendizagem. Para as regras de derivação, serão apresentadas as modelagens e as tabelas resultantes de forma generalista, para entendimento de como os dados ficariam em tabelas reais. Podemos considerar que é como uma engenharia reversa.

A CARDINALIDADE E OS RELACIONAMENTOS NO MODELO LÓGICO

Para as regras de derivação o entendimento de cardinalidade do modelo conceitual deve estar muito bem fixado e, apesar deste assunto já ter sido tratado anteriormente, é muito importante reforçá-lo, pois a derivação do modelo conceitual em lógico depende quase que exclusivamente das cardinalidades.

Na modelagem conceitual a cardinalidade pode ser dada por quantos elementos de uma entidade referem-se à outra entidade. Toda

entidade que se relaciona com outra deve ter uma cardinalidade e, aqui, é representada em mínimo e máximo. A cardinalic, na modelagem conceitual sempre deve ser representada conforme a figura 1, ou seja, um parêntese, a cardinalidade mínima, uma "vírgula", a cardinalidade máxima e o parêntese de fechamento.

[MAAR1] Texto em vermelho não está no resumo cadastrado

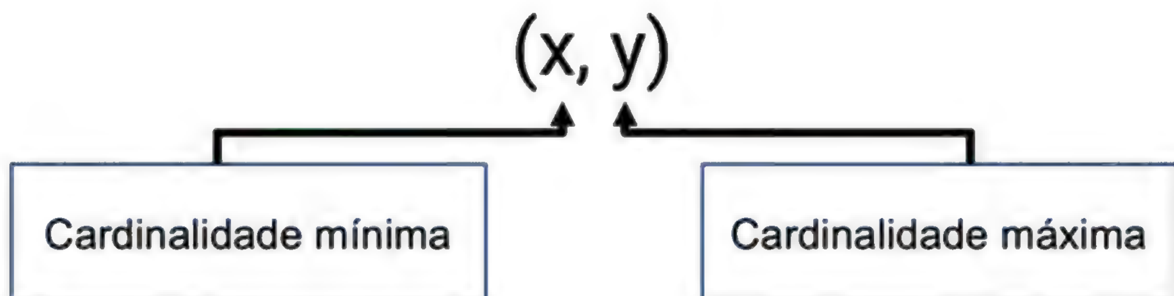


Figura 6 - Representação da cardinalidade no modelo conceitual

Cada par de cardinalidade possui uma regra de derivação específica e, para o fazer, é preciso relembrar, também o conceito de chave estrangeira, FK.

As chaves estrangeiras são importantes pois são elas que representarão os relacionamentos entre as entidades (que agora começaremos a chamar de tabelas). Sempre que estamos derivando o modelo conceitual para o lógico, teremos uma série de chaves estrangeiras em nosso modelo.

Conceito importante sobre chave estrangeira: Todas as colunas marcadas como "FK", sempre apontam para uma coluna de chave primária (*primary key* – PK), ou seja, os dados que são inseridos dentro das colunas do tipo "FK" são dados que apontam para outros dados, sempre identificadores (chave) de alguma outra entidade

(relacionada) com a tabela que está recebendo essa FK ou, da própria tabela, em casos de auto relacionamento.

Este conceito é muito importante pois as cardinalidades influenciarão diretamente em onde as FKs serão colocadas na modelagem lógica e é exatamente isso que descrevem as regras de derivação.

CHAVES ÚNICAS E NÃO NULAS

Antes de apresentarmos como é realizada a derivação do modelo conceitual para o lógico, vamos aprender sobre alguns conceitos muito importantes utilizados em bancos de dados relacionais.

A chave primária (*Primary* Key – PK) nós já aprendemos e podemos considerar que são as que identificam um determinado registro, ou seja, os dados não podem ser repetir (devem ser únicos) e nunca podem ser nulos. Por exemplo, o CPF de uma pessoa, o registro acadêmico de um aluno, o ISBN de um livro, o CRM de um médico etc. Quando a entidade não possui um atributo que possa ser considerado identificador, nós podemos definir um e normalmente o fazemos com o nome “id” (de identificador).

Temos, agora, que entender um novo conceito o da chave **não nula**, ou seja, a chave que chamamos de “NOT NULL”. A chave não nula é aquela cujo dado nunca pode ser vazio, mas não confunda com a chave primária. Aqui os dados podem se repetir. Por exemplo, quando há uma cardinalidade mínima de 1, obrigatoriamente o dado precisará estar presente. Para ilustrar este conceito, pense na relação de autor e livro. Um livro sempre precisará de um autor, então a chave dessa relação sempre precisa ser “não nula” ou, simplesmente, “NOT NULL”.

Existe agora uma quarta chave importante: a chave única ou, chamada simplesmente de "UNIQUE". Essa chave é utilizada quando determinado dado em uma coluna não pode se repetir, mas ao mesmo tempo ele não é chave. A restrição UNIQUE não é chave primária. Por exemplo, o e-mail de um usuário no sistema. O e-mail normalmente não é identificador em uma base de dados, mas também não pode se repetir em usuários diferentes.

Podemos considerar, ainda que a PK é uma chave que é ao mesmo tempo "UNIQUE" e "NOT NULL", para melhorar o entendimento. Podemos agora partir para a derivação da modelagem conceitual.

REGRAS DE DERIVAÇÃO

Podemos classificar os relacionamentos quando aos tipos, baseado em suas cardinalidades. Há, essencialmente sete tipos, embora diferentes autores da área definem isso de formas distintas. Relacionamentos binários 1:1, binários N:N, relacionamentos com atributo identificados, relacionamentos ternários, auto relacionamento 1:1, auto relacionamento 1:N e auto relacionamento N:N. Alguns autores dizem que há apenas três (um para um, um para muitos e muitos para muitos), mas vamos explorar da forma mais completa possível este conceito.

Para ilustrar este conceituo serão apresentadas as modelagens possíveis e suas respectivas tabelas de exemplo e suas modelagens lógicas. Isso também ajudará muito no entendimento do processo de derivação dos modelos. A tabela 1 mostra como o significa das abreviações que serão utilizadas aqui.



ABREVIAÇÃO	EXPLICAÇÃO
A, B, C	Entidades e Tabelas
X, Y, X, W	Atributos nas entidades, chaves primárias e/ou estrangeiras nas tabelas
PK	Primary Key (chave primária)
FK	Foreign KeY (chave estrangeira)

Tabela 5 - Abreviações dos modelos usados para exemplos

Baseado nisso, podemos começar a derivar os modelos e isso será feito com exemplos, para melhor entendimento das regras de derivação

Relacionamentos binários 1:1: Estes são os relacionamentos cuja cardinalidade máxima sempre é "1". A figura 3 ilustra esses relacionamentos as suas respectivas derivações. Para relacionamentos deste contexto, podemos considerar que as regras são:

a) A (1, 1) <-> (1, 1) B à 1 para 1 em ambos os lados sempre deve gerar a **junção** das tabelas

b) A (0, 1) <-> (1, 1) B à Tabela A recebe a FK que aponta para B. Essa FK não pode ser nula

c) A (1, 1) <-> (0, 1) B à Tabela B recebe a FK que aponta para A. Essa FK não pode ser nula

d) A (0, 1) <-> (0, 1) B à Neste caso é indiferente qual das tabelas receberá a FK. Apenas uma deve receber e não há

restrição de nulidade.



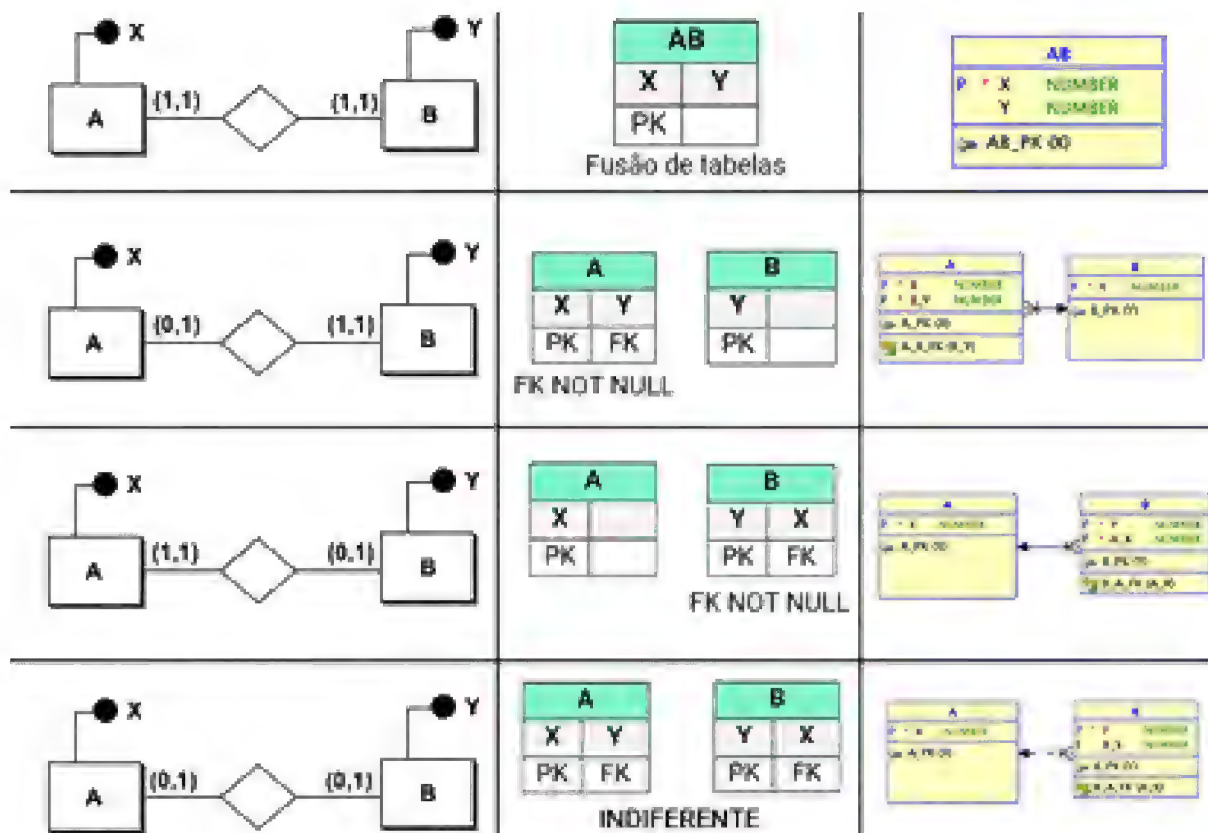


Figura 8 - Derivação em relacionamentos binários 1:1

Relacionamentos binários 1:N: Estes relacionamentos são aqueles cuja cardinalidade máxima em um dos lados é "1" e do outro lado é "N", variando apenas a cardinalidade mínima. A figura 4 ilustra esses relacionamentos as suas respectivas derivações nas modelagens. Para este relacionamento, a entidade que está do mesmo lado que o "N", sempre receberá a FK. Para relacionamentos deste contexto, podemos considerar que as regras são:

a) A(1, 1) <-> (1, N) B à A tabela B recebe a FK que não pode ser nula

b) A(1, 1) <-> (0, N) B à A tabela B recebe a FK que não pode ser nula

c) A(0, 1) <-> (1, N) B à A tabela B recebe a FK que pode ser nula



d) A(0, 1) <-> (0, N) B à A tabela B recebe a FK que pode ser nula

Note que nas relações (1, N) e (0, N) as derivações são iguais e, portanto, podemos considerar que essas relações quando ocorrem são indiferentes para a modelagem lógica, mas muito importantes para a modelagem conceitual.

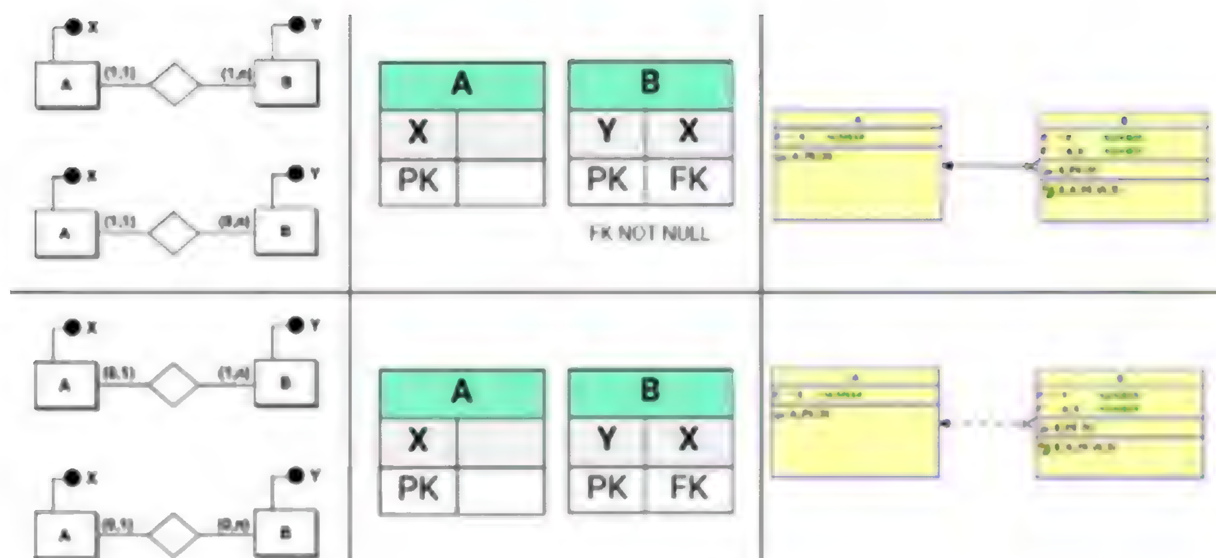


Figura 9 - Derivação de relacionamentos binários 1:N

Relacionamentos binários N:N: Estes relacionamentos ocorrem quando a cardinalidade máxima é "N" (vários) de ambos os lados. Neste caso, sempre é gerada uma tabela nova entre as entidades, chamada de tabela de relacionamento. Essas tabelas de relacionamento no modelo conceitual são as entidades associativas que podem ou não ter atributos de relacionamento (com atributos serão apresentados a seguir), ou seja, podem expressar mais detalhes do relacionamento na própria associação. A figura 5 exemplifica isso de forma genérica para a derivação (1, N) de ambos

os lados. Para as cardinalidades mínimas, basta retirar a obrigatoriedade do campo na tabela de relacionamento. As regras de derivação podem ser descritas como:

- A(1, N) <-> (1, N)B à Gerada uma tabela de relacionamento onde ambos as FKs são "não nulas)

- A(1, N) <-> (0, N)B à Gerada uma tabela de relacionamento. A FKs que aponta para a tabela B deve ser "não nula"

- A(0, N) <-> (1, N)B à Gerada uma tabela de relacionamento. A FKs que aponta para a tabela A deve ser "não nula"

- A(0, N) <-> (0, N)B à Gerada uma tabela de relacionamento. Ambas as FKs não precisam da obrigatoriedade, ou seja, podem ser nulas

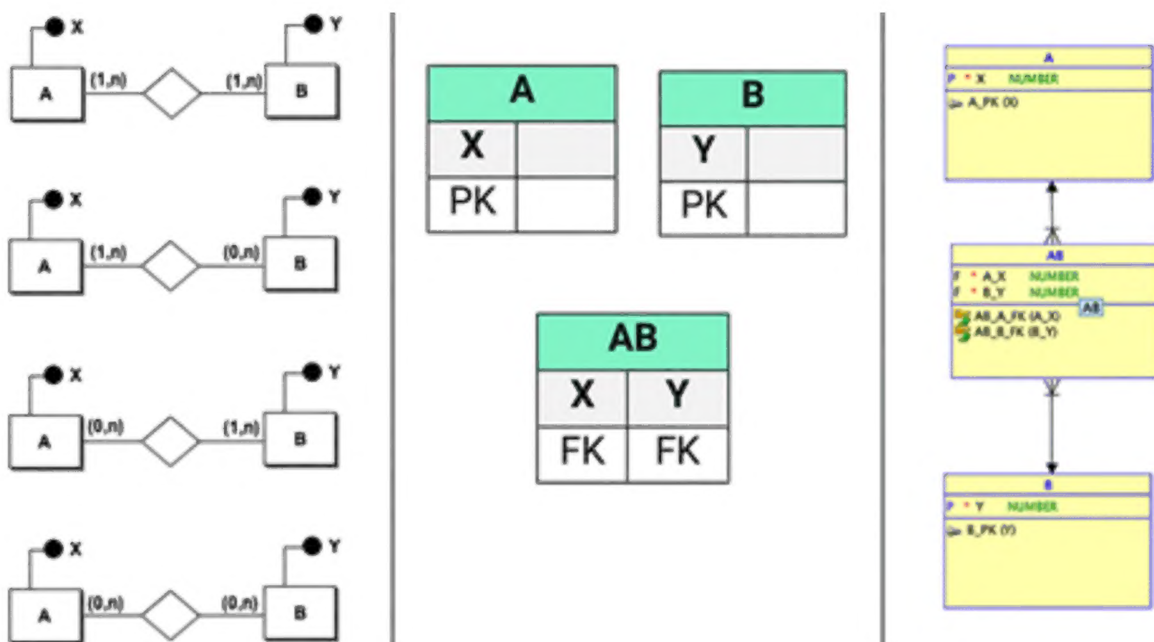


Figura 10 - Derivação de modelos n para n

Relacionamentos binários N:N com dados: Ocorrem quando há uma relação “N” de ambos os lados e na relação há dados que podem ou não ser identificadores. Da mesma forma que os relacionamentos “N para N”, é gerada uma tabela de relacionamento, mas com uma coluna a mais, representando os atributos da relação, conforme pode ser observado na figura 6. Lembre-se que para isso, na modelagem conceitual o relacionamento torna-se uma entidade associativa.

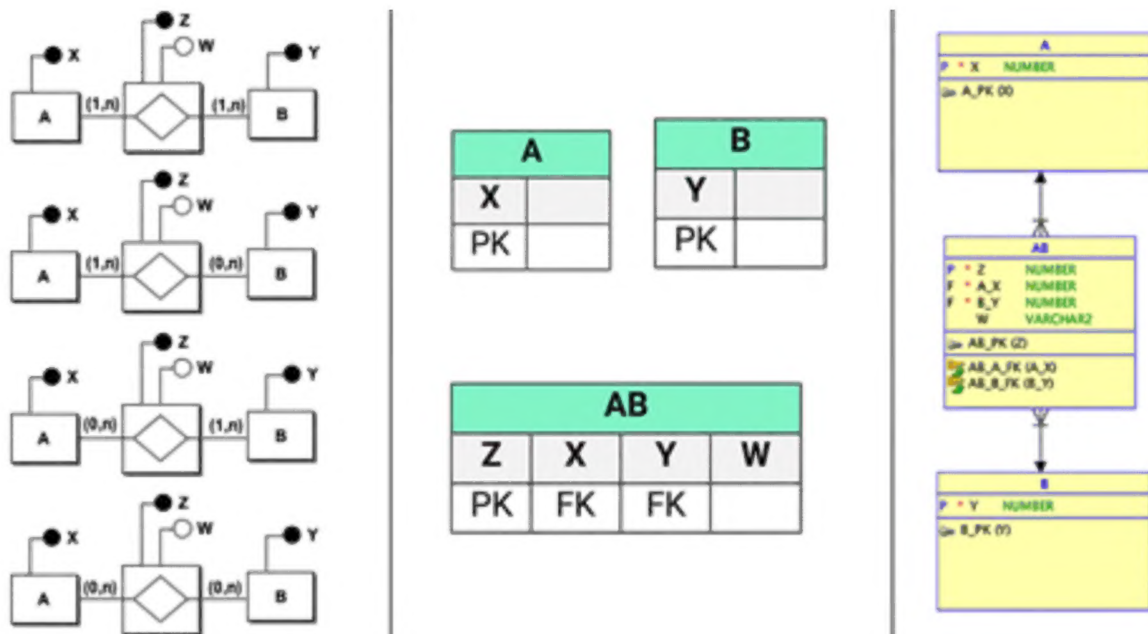


Figura 11 - Derivação de uma relação N para N com dados

Autorelacionamento 1:1 ou 1: N: Neste caso, quando uma entidade se relaciona com ela mesma chamamos de autorelacionamento. A única diferença entre o auto relacionamento 1:1: e 1:N é que no caso da 1:1 a FK dever ser única, ou seja, “UNIQUE”, conforme descrito anteriormente. A figura 7 mostra ambos os exemplos. Um exemplo bastante comum de autorelacionamento é o “casamento” ou a lotação de um departamento, sobre a perspectiva de hierarquia. O exemplo da figura 7 mostra a derivação com dados destes exemplos.

Provavelmente você deve estar se perguntando: Nossa, quantas regras. Precisarei decorar tudo isso? A resposta é simples. Não, não

é necessário, pois uma vez compreendidas as modelagens você saberá realizar as derivações rapidamente, sem ter que ficar consultando todas essas regras. Elas são apresentadas pois fazem parte do processo de ensino e devem sim ser formalizadas. Além do mais, você se acostumará, de tanto usar. Lembre-se, também, que a ideia da engenharia reversa, ou seja, pensar em como as tabelas ficariam fisicamente, pode ajudar bastante no processo. Por isso as tabelas resultantes são apresentadas, apesar de não fazerem parte da modelagem de dados.

Atividade extra

Tente pegar um jogo, um aplicativo ou programa que você utiliza muito e criar a modelagem conceitual dele e, em seguida, demonstrar as tabelas que podem ser geradas e sua respectiva modelagem lógica.

Referência Bibliográfica

- DATE, C. J. Introdução a sistemas de banco de dados. Rio de Janeiro. Ed. Campus, 1991.
- CHEN, Peter. Modelagem de dados: a abordagem entidade-relacionamento para projeto lógico. São Paulo: Makron Books, 1990.
- MEDEIROS, L. F., Banco de dados, princípios e práticas, 1ª. ed., Ed. Intersaberes, 2013

- PUGA, S., França E., GOYA M., Banco de dados: Implementação em SQL, PL/SQL e Oracle 11g, Ed. Pearson, 2013

- ELMASRI R., NAVATHE, S., Sistemas de Banco de Dados, 4ªed., Ed. Pearson, 2005

Ir para exercício